العد و الأعداد في الكومبيوتر

mushtaq_talib58@yahoo.com

أعداد المبرمج : مشتاق طالب رشيد العامري

خبير برامجيات و نظم أمن المعلومات

الراعي الرسمي / شركة الأميرال للبرامجيات و الحلول التقنية

٢٠١٠

المقدمة

اعتاد الإنسان على نظام العد العشري لأنه كان يملك عشرة أصابع في يديه، فعندما يريد إحصاء الأشياء أمامه فكان يقابل كل عنصر من الموجودات أمامه بإصبع واحدة من يديه، و عندما تنتهي أصابع يديه فإنه يحتاج إلى شخص آخر يرفع إصبع واحدة حيث تمثل كل إصبع من أصابع الشخص الثاني عشرة أصابع من أصابع الشخص الأول و بذلك كان الثاني يلعب دور العشرات أما الأول فيلعب دور الآحاد.

و بعد اختراع الكتابة سارع علماء الرياضيات إلى تحويل نظام العد العشري إلى صيغة كتابية، فاعتمدوا الأساس التالي: (نمثل الأعداد من ١ حتى ٩ برمز واحد فقط أما العدد الذي يأتي بعد التسعة فهو عبارة عن مزيج رمزين الأول هو الصفر و الثاني هو الواحد).

من الفكرة السابقة نجد أن الرموز الأساسية لنظام العد العشري هي من الصفر حتى التسعة أي هي عشرة رموز نستطيع من خلالها تكوين عدد أي عدد طبيعي.

طريقة العد:

نبدأ بالعد اعتباراً من أول رمز و هو الصفر و نزيد بمقدار واحد واحد إلى أن نصل إلى نهاية الرموز ألا و هو التسعة، و إذا أردنا المتابعة فإننا نصفر الخانة التي نتعامل معها و نضيف واحد إلى الخانة المجاورة لنحصل على الرقم عشرة (١٠) و من ثم نبدأ بزيادة الآحاد من جديد حتى نصل إلى ١٩ عندها نصفر الآحاد و نضيف واحد إلى خانة العشرات فينتج العدد ٢٠ و هكذا حتى نصل إلى العدد ٩٩ عندها نحاول زيادة خانة الآحاد فلا نستطيع فنصفرها و نحاول زيادة العشرات فلا نستطيع أيضاً فنصفرها و نزيد خانة إلى منزلة المئات فنحصل على العدد ١٠٠.

العد بالنظام الست عشري

يختلف هذا النظام عن سلفه بأن الرموز الأساسية هي من الصفر حتى التسعة و يأتي بعد التسعة الأحرف من A حتى F أي أن الرموز الأساسية هي:

{ ٠,١,٢,٣,٤,٥,٦,٧,٨,٩,A,B,C,D,E,F }

للأتصال/ mushtaq_talib58@yahoo.com

و لكي تستطيع العد بسهولة في هذا النظام أعد قراءة التمهيد و لكن تخيل جدلاً أن للإنسان ست عشرة إصبع في كل يد ثمانية أصابع !!

تمرين على العد بالنظام الست عشري:

٠,١,٢،...،٩،
A,B,...,F,10,11,12,13,14,...,19,1A,1B,1C,...,1F,20,21,...,29,2A,2B,...
,2F,30,...,99,9A,9B,...,9F,A0,A1,A2,...,A9,AA,AB,AC,...,AF,...,FF,10
0,...,119,11A,11B,...,199,19A،...

نظام العد الثنائي

تتطلب أجهزة الحواسيب و الأجهزة الإلكترونية نظام عد جديد ملائم لطبيعة هذه الأجهزة، فنحن نعلم أن جميع الأجهزة الإلكترونية تعمل على التيار الكهربائي و الذي له حالتين هما الوضع on و الوضع off .

و بذلك كان النظام الثنائي هو الحل حيث اعتمد على رمزين فقط في تمثيل أعداده هما الصفر و الواحد {٠,١} .

العد بالنظام الثنائي:

٠٠٠٠,٠٠٠١,٠٠١٠,٠٠١١,٠١٠٠,٠١٠١,٠١١٠,٠١١١,١٠٠٠,١٠٠١,١٠١٠
١٠١١,١١٠٠,١١٠١,١١١٠,١١١١،

التحويل بين نظم الأعداد

يلزمنا في لغة الأسمبلي التحويلات التالية:

١- التحويل من الثنائي إلى العشري.

٢- التحويل من الست عشري إلى العشري.

٣- التحويل من العشري إلى الثنائي.

و سنعطي مثالاً عن كل حالة من هذه الحالات:

مثال١ : حول الرقم الثنائي التالي ٠١٠٠ إلى مقابله في نظام العد العشري:

مثال٢ :حول العدد الست عشري التالي A٣٣ إلى مقابله في نظام العد العشري:

مثال٣ : حول العدد العشري التالي ٣٠ إلى مقابله في النظام الثنائي:

لدينا الجدول المرسوم جانباً:

نستخدم هذا الجدول من أجل هذا النوع من التحويل فلتحويل العدد العشري ٣٠ نلاحظ أنه مكون من ١٦+٨+٤+٢ فنضع واحدات تحت الأعداد ١٦ و ٨ و ٤ و ٢ و نملأ الباقي أصفاراً، و بذلك نحصل على الرقم الثنائي المقابل.

المتمم الثنائي و كيفية الحصول عليه

يستخدم المتمم الثنائي من أجل تمثيل الأعداد السالبة في الحاسب في النظام الثنائي و لتمثيل عدد سالب نتبع الخطوات التالية:

١- نكتب العدد بالنظام الثنائي.

٢- نقلب الأصفار واحدات و الواحدات أصفاراً.

٣- نضيف واحد إلى الرقم الناتج.

مثال: مثّل العدد ٣٠- بالنظام الثنائي عن طريق المتمم الثنائي:

(٣٠) d= 0001 1110

نقلب => ٠٠٠١ ١١١٠

نضيف => ٠٠١٠ ١١١٠

لمحة عن الحاسب

يُعرَّف الحاسب الرقمي بأنه نظام إلكتروني لمعالجة المعطيات، و يتألف من قسمين أساسيين:

للأتصال/ mushtaq_talib58@yahoo.com

القسم الأول : البرمجيات SOFTWARE .

القسم الثاني : الكيان الصلب HARDWARE

و يقسم الكيان الصلب إلى أقسام رئيسية هي :

١) وحدة الدخل: تتم من خلالها إدخال المعطيات الرقمية.

٢) وحدة الخرج: تتم من خلالها إظهار النتائج بعد معالجة المعطيات.

٣) وحدة المعالجة المركزية: هي المسؤولة عن العمليات الحسابية و المنطقية و معالجة البيانات.

٤) وحدة الذاكرة: تخزن البرامج و المعطيات.

البنية الداخلية للمعالج ٨٠٨٦

للأتصال/ mushtaq_talib58@yahoo.com

يتألف المعالج ٨٠٨٦ من وحدتين منفصلتين هما :

١) وحدة ملائمة الممرات (Execution Unit) : و سنرمز لها بالرمز EU .

٢) وحدة التنفيذ (Bus interface Unit) : و سنرمز لها بالرمز BIU .

بشكل عام فإن الـ BIU مسؤولة عن معظم الأعمال مثل : إحضار التعليمة، قراءة و كتابة المتحولات في الذاكرة، إدخال و إخراج المعطيات من و إلى الأجهزة المحيطية.

أما الـ EU فهي مسؤولة عن تنفيذ التعليمات. و كلا الوحدتين تعملان بشكل متوازٍ لتخفيض الزمن المطلوب لإحضار عدة تعليمات و تنفيذها.

ملاحظة: من الجدير بالذكر بأن هنالك ثلاثة ممرات في الحاسب و هي:

١) ممر المعطيات DATA BUS : و يصل بين المعالج و الذاكرة وظيفته نقل المعطيات من و إلى الذاكرة.

٢) ممر العناوين ADDRESS BUS : و يصل بين المعالج و الذاكرة أيضاً و وظيفته نقل العناوين من المعالج إلى الذاكرة.

٣) ممر التحكم CONTROL BUS : لتنسيق عمل الممرين السابقين.

وحدة ملائمة الممرات Bus Interface Unit

و تستخدم لملائمة المعالج مع العالم الخارجي. و تتألف من : جامع العناوين، مسجلات المقاطع، وحدة التحكم بالمحرف، صف التعليمات.

تقوم وحدة الـ BIU بالتحكم بممر المعطيات و ممر العناوين و ممر التحكم .

تحضر BIU التعليمات من الذاكرة بايت بايت و تضعها فيما يسمى برتل التعليمات (صف التعليمات) الذي يتسع لست بايتات كحد أعظمي و من الطبيعي أن التعليمة التي تدخل رتل التعليمات أولاً يتم تنفيذها أولاً للمحافظة على ترتيب التعليمات و يدعى هذا المبدأ بـ الداخل أولاً خارج أولاً First In Last Out و نرمز لهذا المبدأ بـ FIFO.

إن إحضار شيفرة التعليمة التالية يتم عندما تكون وحدة التنفيذ EU مشغولة بتنفيذ التعليمة الحالية (هذه إحدى محسنات المعالج ٨٠٨٦ عن أسلافه حيث كانت الـ CPU في المعالجات السابقة للمعالج ٨٠٨٦ تتوقف عن العمل خلال فترة تنفيذ التعليمة الحالية).

عندما تفك وحدة التنفيذ EU شيفرة تعليمة ما من رتل التعليمات و تكون هذه التعليمة تعليمة تؤدي إلى تغيير تسلسل تعليمات البرنامج (قفز إلى برنامج فرعي مثلاً) عندها يتم تصفير رتل التعليمات و إعادة ملئه من جديد بتعليمات البرنامج الفرعي (لأن وحدة ملائمة الممرات BIU تجلب التعليمات دون معرفة ما تؤديه هذه التعليمات).

ملاحظة: جامع العناوين و مسجلات المقاطع سيتم شرحها لاحقاً.

وحدة التنفيذ Execution Unit

و هي مسؤولة عن فك شيفرة التعليمات و تنفيذها و تتألف من :

١) و حدة الحساب و المنطق.

٢) مسجل الأعلام.

٣) ثمانية مسجلات للأغراض العامة.

٤) مسجلات مؤقتة.

٥) منطق التحكم بـ EU.

للأتصال/ mushtaq_talib58@yahoo.com

تجلب وحدة التنفيذ EU التعليمات من مقدمة رتل التعليمات في وحدة ملائمة الممرات BIU و تفك شيفرتها و تقوم بالعمل الذي تمليه كل تعليمة فإذا احتاجت هذه الوحدة (EU) إلى معلومة مخزنة في الذاكرة فإنها تأمر وحدة ملائمة الممرات BIU بإحضارها و ذلك عن طريق إعطائها عنوان هذه المعلومة في الذاكرة.

إن من أحد أهم وظائف EU هو تنفيذ العمليات الحسابية و المنطقية على المعلومات، و أثناء سير التنفيذ تقوم EU بفحص مسجل الأعلام بعد كل تعليمة (مسجل الأعلام : هو عبارة عن ستة عشر بت تعبر عن حالة المعالج بعد تنفيذ كل تعليمة) .

مسجلات الأغراض العامة هي ثمانية مسجلات طول كل مسجل منها ٢ بايت و هذه المسجلات هي AX,BX,CX,DX,SI,DI,BP,SP .

بنية الذاكرة

تتألف الذاكرة من حجرات متسلسلة سعة كل منها ٨ بت (واحد بايت) ، ترقم هذه الحجرات من الصفر و حتى نهاية الذاكرة و يستخدم النظام الست عشري عادة في عملية الترقيم و بذلك يكون لكل حجرة رقم يميزها عن غيرها، يدعى هذا الرقم بعنوان تلك الحجرة.

يوضع داخل كل حجرة رقم ست عشري يتراوح بين ٠ و FF و يدعى هذا الرقم بمحتوى تلك الحجرة.

يوجد بين المعالج و الذاكرة ممران هما ممر المعطيات بعرض ١٦ بت و ممر العناوين بعرض ٢٠ بت.

فمثلاً عندما يحتاج المعالج إلى القيمة المخزنة في الحجرة ذات الرقم ١٠٠ (عنوانها ١٠٠) فإن الرقم ١٠٠ يمثل بشكل ثنائي و يوضع على ممر العناوين و يرسل إلى الذاكرة، و حالما تستلم الذاكرة هذا العنوان فإن محتوى الحجرة ١٠٠ يرسل إلى المعالج عن طريق ممر المعطيات.

إن كون ممر العناوين ذو عرض ٢٠ بت (٢٠ خط نقل) هذا يعني أنه يستطيع نقل رقم ثنائي ذو ٢٠ خانة أي أن أكبر قيمة يمكن وضعها على ممر العناوين هي :

للأتصال/ mushtaq_talib58@yahoo.com

و بذلك يستطيع المعالج ٨٠٨٦ عنونة واحد ميغا من الذاكرة فقط.

مقاطع الذاكرة (هذه الفقرة مرتبطة ارتباطاً وثيقاً بالمسجلات)

يتعامل المعالج كما ذكرنا مع واحد ميغا من الذاكرة، و يمكن أن نقتطع من هذه الميغا أربعة مقاطع أساسية يتعامل معها برنامجنا بشكل مباشر (أي أنه لا تتم الاستفادة من كل الذاكرة بآن واحد) و هذه المقاطع الأربعة هي:

١) مقطع الشيفرة Code Segment CS

يخصص هذا المقطع من الذاكرة –كما هو واضح من تسميته- لتخزين شيفرة البرنامج. و هناك مسجل له نفس الاسم CS موجود في المعالج يحتفظ بقيمة تدل على بداية هذا المقطع في الذاكرة و يساعده المسجل IP(Instruction Pointer) الذي يحتفظ بعنوان التعليمة التي ستنفذ الآن و تعدل قيمته آلياً ليشير إلى عنوان التعليمة التالية.

٢) مقطع المعطيات Data Segment DS

يخصص هذا المقطع من الذاكرة لتخزين المعطيات و المتحولات. و هناك مسجل له نفس الاسم DS موجود في المعالج يحتفظ بقيمة تدل على بداية هذا المقطع في الذاكرة و يساعده المسجل SI الذي يشير إلى الإزاحة بالنسبة إلى بدايته.

٣) مقطع المكدس Stack Segment SS

يخصص هذا المقطع للحفظ المؤقت لبعض المعلومات الضرورية و التي يخشى أن تضيع أو تتغير أثناء تنفيذ برنامج ما. و هناك مسجل له نفس الاسم SS موجود في المعالج يحتفظ بقيمة تدل على بداية هذا المقطع في الذاكرة.

آلية عمل المكدس Last In First Out LIFO (آخر ما يدخل أول ما يخرج) : أي أن أول عنصر يدخل إلى المكدس يصبح في قعره و آخر عنصر يدخل المكدس يصبح في قمته و يتم سحب المعلومات من المكدس من قمته حيث لدينا مسجل اسمه Stack Pointer SP يشير دوماً إلى قمة المكدس فهو يتغير حسب الحالة التي يتم

بها التعامل مع المكدس (إدخال معلومات أو إخراج). فعند إدخال معلومة بطول ٢ بايت فإن قمة المكدس تقترب من بداية الذاكرة (انظر الشكل) و بذلك تنقص قيمة SP بمقدار ٢ لأن إملاء المكدس يعني الاقتراب من العنوان الأصغر و العكس بالعكس أي عندما نسحب معلومة من المكدس فإن قمته تبتعد عن بداية الذاكرة و بذلك تزيد SP بمقدار ٢ لأن إفراغ المكدس يعني الاقتراب من العنوان الأكبر.

٤) مقطع المعطيات الإضافي Extra Segment ES

يستخدم عند الحاجة إلى استخدام مقطعي معطيات بنفس الوقت و بذلك نستطيع الاستفادة من مساحة أكبر في الذاكرة. و يساعده المسجل Destination Index DI الموجود في المعالج و الذي يشير إلى الإزاحة بالنسبة إلى بدايته.

ملاحظة: يجب التمييز بين المقطع و مسجل المقطع حيث المقطع هو جزء من الذاكرة بينما مسجل المقطع يتألف من بايتين و هو موجود في المعالج.

المسجلات Registers

يملك المعالج ٨٠٨٦ أربعة مجموعات من المسجلات ذات ١٦ بت يستطيع المبرمج الوصول إليها و هي:

١) مؤشر التعليمة IP

٢) أربعة مسجلات معطيات AX,BX,CX,DX .

٣) أربعة مسجلات تأشير و فهرسة SI,DI,BP,SP .

٤) أربعة مسجلات مقاطع CS,DS,SS,ES .

بالإضافة إلى ذلك يوجد مسجل آخر هو مسجل الأعلام و يدعى أيضاً مسجل الحالة و هو مسجل ذو ١٦ بت و لكن نستخدم منه ٩ خانات فقط.

سنشرح كلٍ من هذه المسجلات بالتفصيل :

المجموعة الأولى : مسجلات المقاطع

و هي عبارة عن أربعة مسجلات طول كل منها ١٦ بت أي ٢ بايت و هي :

١) مسجل مقطع الشيفرة CS : يحتوي على عنوان أول حجرة في مقطع شيفرة البرنامج في الذاكرة، أي أنه يشير إلى بداية مقطع الشيفرة.

٢) مسجل مقطع المعطيات DS : يحتوي على عنوان أول حجرة في مقطع المعطيات في الذاكرة، أي أنه يشير إلى بداية مقطع المعطيات.

٣) مسجل مقطع المكدس SS : يحتوي على عنوان أول حجرة في مقطع المكدس في الذاكرة، أي أنه يشير إلى بداية مقطع المكدس.

٤) مسجل مقطع المعطيات الإضافي ES : يحتوي على عنوان أول حجرة في مقطع المعطيات الإضافي في الذاكرة، أي أنه يشير إلى بداية مقطع المعطيات الإضافي.

المجموعة الثانية: مسجلات الفهرسة و التأشير

و هي عبارة عن أربعة مسجلات مساعدة تساعد في إيجاد العنوان الفيزيائي بالتعاون مع مسجلات المقاطع، و طول هذه المسجلات ١٦ بت أي ٢ بايت، و هي :

١) مسجل دليل المصدر Source Index SI : يخزن فيه عنوان يدل على الإزاحة ضمن مقطع المعطيات DS و بمعنى آخر يستعمل في إمساك العناوين الفعالة من أجل التعليمات التي تتناول المعطيات المخزنة في مقطع المعطيات في الذاكرة.

٢) مسجل دليل الهدف Destination Index DI : يخزن فيه عنوان يدل على الإزاحة ضمن مقطع المعطيات الإضافي ES ، و بمعنى آخر يستعمل مسجل دليل الهدف DI من أجل استنتاج العنوان الفيزيائي الذي يحدد حجرة متحول الهدف.

٣) مسجل مؤشر المكدس Stack Pointer SP : يسمح مؤشر المكدس بوصول سهل للحجرات في مقطع المكدس الموجود في الذاكرة حيث أن القيمة في SP تمثل العنوان الفعال لحجرة المكدس التالية التي يمكن الوصول إليها نسبة إلى العنوان الحالي الموجود في مسجل مقطع المكدس SS و يحتفظ SP دوماً بقيمة تدل على قمة

المكدس ، هذا و إن قيمة هذا المسجل تتعدل تلقائياً عند وضع أو سحب معلومة بالمكدس.

٤) مسجل مؤشر القاعدة Base Pointer BP : يحوي قيمة تدل على الإزاحة بالنسبة لمقطع المكدس SS و هو يستخدم لقراءة المعطيات ضمن مقطع المكدس بدون إزالتها من المكدس.

المجموعة الثالثة: مسجلات المعطيات

تستخدم هذه المسجلات من أجل التخزين المؤقت للنتائج المرحلية أثناء تنفيذ البرنامج حيث أن تخزين المعطيات في هذه المسجلات يمكننا من الولوج إلى تلك المعطيات بشكل أسرع مما لو كانت في الذاكرة، و تقسم المسجلات إلى :

١) مسجل المراكم Accumulator و يرمز له بالرمز A .

٢) مسجل القاعدة Base و يرمز له بالرمز B .

٣) مسجل العد Count و يرمز له بالرمز C .

٤) مسجل المعطيات Data و يرمز له بالرمز D .

و كل مسجل من المسجلات السابقة يمكن استعماله إما ككلمة ١٦ بت و يدل على ذلك بكتابة الحرف X بعد اسم المسجل أو يمكن استعماله كبايتين كل منهما ٨ بت و يدل على ذلك باستخدام الحرفين H,L حيث :

L للبايت ذو العنوان الأصغر ، مثال AL .

H للبايت ذو العنوان الأكبر ، مثال BH .

هكذا و إن كلاً من هذه المسجلات يمكن استخدامه من أجل التعليمات الرياضية أو المنطقية في لغة الأسمبلي مثل And, Add .

و من أجل بعض التعليمات مثل البرامج التي تحتوي على تعليمات سلاسل فإنها تستعمل مسجلات معينة مثل استعمال المسجل C لتخزين العدد الذي يمثل عدد البايتات التي ستنفذ عليها تعليمات السلاسل (عدد مرات تكرار تعليمة السلسلة)

مسجل مؤشر التعليمة Instruction Pointer IP

هذا المسجل يحدد موقع التعليمة التالية التي ستنفذ في مقطع الشيفرة و بعد جلب شيفرة التعليمة من الذاكرة فإن BIU تعدل قيمة IP بحيث تشير إلى التعليمة التالية في الذاكرة (التعديل يتم آلياً).

مسجل الأعلام Flags Register

هو مسجل ذو ١٦ بت موجود في وحدة التنفيذ كما هو واضح بالشكل :

٠	١	٢	٣	٤	٥	٦	٧	٨	٩	١٠	١١
	١٢	١٣	١٤	١٥							
CF		PF		AF		ZF	SF	TF	IF	DF	OF

و كما نلاحظ من الشكل السابق أنه يوجد ستة أعلام للحالة هي CF, PF, AF, ZF, SF, OF ، و كذلك يوجد ثلاثة أعلام للتحكم DF, IF, TF .

أ) أعلام الحالة

تشير إلى الحالات الناتجة كنتيجة لتنفيذ تعليمة منطقية أو رياضية حيث تكون إما في حالة واحد منطقي Set أو تكون في حالة صفر منطقي Reset ، و سنلخص فيما يلي عمل كلٍ منها:

أولاً: علم الإنزياح Carry Flag

يكون في حالة الواحد المنطقي إذا وجد انزياح خارجي (حمل) أو استعارة من أجل الخانة الأخيرة (البت الأخير) و ذلك أثناء تنفيذ التعليمات الرياضية.

و يكون في حالة الصفر المنطقي إذا لم يوجد حمل أو استعارة من أجل البت الأخير.

أمثلة:

أولاً: حالة الإنزياح

للأتصال/ mushtaq_talib58@yahoo.com

٤ ٣ ٢ ١ ٠

٧ ٦ ٥

٠ ١ ١ ٠

١ ١ ٠ ٠

٠ ١ ١ ١ +

١ ١ ٠ ٠

٠ ١ ١ ٠ ١

١ ٠ ٠

ثانياً: حالة الاستعارة

٤ ٣ ٢ ١ ٠

٧ ٦ ٥

١ ٠ ١ ١

٠ ١ ٠ ١

١ ٠ ٠ ٠ -

١ ١ ١ ١

٠ ٠ ١ ١

٠ ١ ١ ٠

و في المثالين السابقين نطبق نفس الكلام من أجل ٢ بايت و لكن الإنزياح الخارج و الاستعارة تكون من أجل البت الخامس عشر (الأخير).

ثانياً: علم الازدواجية Parity Flag PF

يصبح في حالة واحد منطقي إذا كانت نتيجة آخر تعليمة تحوي على عدداً زوجياً من الخانات الواحدية (بعد التحويل إلى النظام الثنائي طبعاً) و إلا يكون في حالة الصفر المنطقي.

نلاحظ أن علم PF يفحص البايت السفلي فقط حتى لو كنا نتعامل مع كلمة (٢ بايت) ، أما عندما نتعامل مع بايت واحد فقط فإنه يفحصه كله.

ثالثاً: علم الإنزياح المساعد Auxiliary Flag AF

يكون في حالة الواحد المنطقي إذا وجد إنزياح من النصف السفلي إلى النصف العلوي أو استعارة من النصف العلوي إلى النصف السفلي و ذلك من أجل البايت السفلي من الكلمة (٢ بايت) و بمعنى آخر أنه إذا كان لدينا إنزياح من الخانة ٣ إلى الخانة ٤ فإن AF=1 و ذلك في حال كانت المعطيات بايت واحد أو بايتين (كلمة)، و فيما عدا ذلك يكون AF=0 .

مثال:

٣ ٢ ١ ٠

٧ ٦ ٥ ٤

٠ ١ ١ ٠

١ ١ ٠ ٠

٠ ٠ ١ ١ ١ +

١ ١ ٠

٠ ١ ١ ٠ ١

٠ ٠ ١

رابعاً: علم الصفر Zero Flag ZF

يصبح في حالة واحد منطقي عندما يكون ناتج آخر عملية حسابية أو منطقية يساوي الصفر.

يصبح في حالة صفر منطقي عندما يكون ناتج آخر عملية حسابية أو منطقية لا يساوي الصفر.

خامساً: علم الإشارة Sign Flag SF

يكون علم SF في حالة واحد منطقي Set إذا كانت نتيجة آخر عملية حسابية عدداً سالباً.

يكون علم SF في حالة صفر منطقي Reset إذا كانت نتيجة آخر عملية حسابية عدداً موجباً.

مصطلح: من إحدى طرق تمثيل الأعداد السالبة في الكمبيوتر هي اعتبار الخانة الأخيرة مخصصة للإشارة و بما أن البايت مكون من ثمانية خانات فسيتم اقتطاع الخانة الأخيرة منه من أجل الإشارة فإن احتوت على القيمة واحد فإن الخانات السبعة الباقية هي عدد ثنائي سالب أما إذا احتوت على القيمة صفر فإن الخانات السبعة المتبقية ما هي إلا عدد موجب.

للأتصال/ mushtaq_talib58@yahoo.com

و بذلك يكون SF هو نسخة عن الخانة الأخيرة في الناتج عند اعتماد هذا النظام لتمثيل الأعداد السالبة.

لاحظ أنه انطلاقاً من هذا المبدأ في التمثيل يمكننا تمثيل المجالات التالية من الأعداد:

من أجل بايت واحد من –١٢٨ إلى +١٢٧

من أجل بايتين من –٣٢٧٦٨ إلى +٣٢٧٦٧

سادساً: علم الطفحان Overflow Flag OF

يكون في حالة واحد منطقي عندما لا تتسع النتيجة في المكان المخصص لتخزينها أي تتجاوز القدرة التخزينية، أما إذا لم تكن النتيجة خارج المجال المحدد فإن OF يبقى في حالة الصفر المنطقي.

يحدث الطفحان في الحالات التالية:

١) جمع أعداد موجبة كبيرة.

٢) جمع أعداد سالبة كبيرة.

٣) طرح عدد موجب كبير من عدد سالب كبير.

٤) طرح عدد سالب كبير من عدد موجب كبير.

ملاحظة: جميع الأعلام السابقة ما عدا CF تُقرأ فقط أي لا نستطيع تغيير محتواها لذلك يمكن قراءتها فقط و لا يمكن تغيير محتوياتها بواسطة تعليمات برمجية مباشرة.

المعالج مزود بتعليمات تستطيع اختبار حالة هذه الأعلام لتغيير تتابع تنفيذ البرنامج فمثلاً يمكن اختبار علم ZF=1 كشرط من أجل القفز إلى جزء آخر من البرنامج.

و فيما يلي سنشرح أعلام التحكم:

أولاً: علم الخطوة الوحيدة Trap Flag TF

يوضع بالحالة واحد منطقي عندما نرغب بتنفيذ البرنامج خطوة خطوة و هو مفيد عندما نريد تصحيح برنامجنا و استكشاف مواقع الأخطاء.

ثانياً: علم المقاطعة Interrupt Flag IF

يستخدم من أجل التعبير عن إمكانية أو عدم إمكانية تنفيذ المقاطعة، فيوضع بالحالة واحد منطقي عندما لا نرغب بتنفيذ أي مقاطعة (المقاطعة محجوبة) أما عند وضعه في حالة الصفر المنطقي فإن المقاطعة مسموح بها.

ملاحظة: المقاطعة هي عبارة عن خدمة تؤدي إلى عمل معين فمثلاً المقاطعة ٢١ و التي من أحد خدماتها العودة إلى نظام التشغيل.

ثالثاً: علم الاتجاه Direction Flag DF

يدل على اتجاه سير العمليات التسلسلية.

عندما يكون في حالة واحد منطقي فإن السلسلة تكون من العنوان الأعلى إلى العنوان الأدنى.

عندما يكون في حالة صفر منطقي فإن السلسلة تكون من العنوان الأدنى إلى العنوان الأعلى.

مفهوم العنوان الفيزيائي و الإزاحات

مقدمة

لاحظنا أن الذاكرة بطول ١ ميغا بايت أي أنها مرقمة من h٠٠٠٠٠ إلى FFFFFh لذلك فإننا نحتاج أثناء عنونة المقاطع إلى رقم ست عشري بطول ٢٠ بت ذلك لأن تمثيل رقم ست عشري بطول خمس خانات (و هو المستخدم في ترقيم حجرات الذاكرة) يحتاج إلى عشرين بت لكن مسجلات المقاطع و التي نستخدمها في العنونة هي بطول ١٦ بت فقط الأمر الذي يضطرنا إلى استنتاج عنوان فيزيائي بعشرين بت !!

آلية الحصول على العنوان الفيزيائي Physical Address PA

يلزمنا لإيجاد العنوان الفيزيائي قيمتين هما :

١) قيمة مسجل المقطع ٢) قيمة المسجل المساعد له

فكرة Very good Tip :

عندما نريد إزاحة رقم ممثل بالنظام العشري خانة واحدة نحو اليسار فإننا نضربه بعشرة !!

للأتصال/ mushtaq_talib58@yahoo.com

مثال: هل تستطيع إزاحة الرقم ١٩٢ إلى اليسار خطوة واحدة ليصبح ١٩٢٠ ؟؟

نعم و ذلك بضربه بعشرة كالتالي ١٩٢ x 10= 1920

و كذلك الأمر في النظام الست عشري، فعندما نريد إزاحة رقم ست عشري فإننا نضربه بعشرة النظام الست عشري و التي هي

١٠h= 16 d

لذلك يتم الحصول على العنوان الفيزيائي بالطريقة التالية:

١) نأخذ قيمة مسجل المقطع الممثلة بالنظام الست عشري و نضربها بعشرة النظام الست عشري فتنزاح قيمة مسجل المقطع خانة واحدة نحو اليسار.

٢) نجمع قيمة المسجل المساعد لنفس المقطع و الممثلة أيضاً بالنظام الست عشري فتكون النتيجة هي حصولنا على العنوان الفيزيائي

(مسجل المقطع x 10h) + قيمة المسجل المساعد = PA(Physical Address)

أمثلة:

بفرض لدينا مسجل مقطع الشيفرة CS يحتوي على B6C١ و قيمة مسجل مؤشر التعليمة IP المساعد له هي A0 أوجد العنوان الفيزيائي للتعليمة :

الحل:

PA= (CS x 10h) + IP= 1B6C x 10h+ A0= 1B760

للأتصال/ mushtaq_talib58@yahoo.com

مثال آخر: أوجد PA بفرض DS= 1000h و SI= 1F .

الحل:

$$PA = (1000 \times 10) + 1F = 1001F$$

الطريقة العكسية (هذه الطريقة يجب إتقانها ذهنياً)

عندما نُعطى العنوان الفيزيائي و نريد استنتاج قيمة مسجل المقطع (عنوان المقطع) و قيمة المسجل المساعد له (الإزاحة) نتبع إحدى الطريقتين التاليتين :

الطريقة الأولى

١- نأخذ الخانات الأربعة اليمينية من العنوان الفيزيائي المعطى و نعتبرها إزاحة (أي نضع قيمتها في المسجل المساعد) .

٢- نصفر الخانات الأربعة الأولى من العنوان الفيزيائي فينتج معنا رقم ست عشري أول أربع خانات منه أصفاراً .

٣- نحذف الصفر الأول من الرقم الناتج فينتج معنا رقم ست عشري هو قيمة مسجل المقطع .

مثال:

بفرض لدينا عدد موجود في العنوان الفيزيائي ٤١٠٠٠h أوجد قيمة مسجل المعطيات DS و قيمة المسجل المساعد له SI .

الحل: حسب الطريقة بأخذ الخانات الأربعة الأولى من على اليمين تكون قيمة SI تساوي ١٠٠٠h و هي الإزاحة.

للأتصال/ mushtaq_talib58@yahoo.com

٢-٣)   4000h = DS

طريقه أخرى

١- نأخذ الخانة الأولى من العنوان الفيزيائي ونعتبرها إزاحة.

٢- نحذف تلك الخانة من العنوان الفيزيائي فيصبح الرقم الناتج مؤلف من أربع خانات و هو يمثل قيمة مسجل المقطع.

مثال: بفرض كان 41000h =PA

الحل : بأخذ الخانة الأولى

١)   0=SI

٢)   4100=DS أخذنا الخانات المتبقية من الرقم

أي أن

٤١٠٠:٠٠٠٠   ٤٠٠٠:١٠٠٠

عنوان إزاحته   عنوان إزاحته

الموديل البرمجي للمعالج ٨٠٨٦

أساليب العنونة

مقدمة:

إن حيز الذاكرة منظم على شكل بايتات معنونة من ٠٠٠٠٠h إلى FFFFFh لذلك من أجل كلمات المعطيات ١٦ بت يتم تخزين البايت السفلي في العنوان الأصغر و البايت العلوي في العنوان الأكبر كما نعلم أن الذاكرة تحتوي أربع مقاطع كل منها ٦٤KB و هي مقطع الشيفرة و مقطع المعطيات و مقطع المكدس و مقطع المعطيات

للأتصال/ mushtaq_talib58@yahoo.com

الإضافي، حيث يتم الرجوع إلى هذه المقاطع بمساعدة مسجلات المقاطع ذات الـ١٦ بت و هي CS, DS, SS, ES و كل من هذه المسجلات يحتوي عنواناً قاعدياً ذا ١٦ بت و الذي يستخدم في توليد العنوان الفيزيائي للذاكرة و الذي يشير إلى بداية المقطع المطابق في الذاكرة.

يستطيع المبرمج تبديل القيم في مسجلات المقاطع برمجياً، فمثلاً : يمكن تهيئة مقطع معطيات جديد ببساطة و ذلك بتبديل قيمة المسجل DS عن طريق تنفيذ التعليمتين التاليتين :

```
Mov AX,A000

Mov DS,AX
```

و سبب وجود هاتين التعليمتين هو عدم وجود تعليمة واحدة لتحميل مسجل مقطع بعدد ثابت.

إن المعالج ٨٠٨٦ مزود بتسعة أنظمة عنونة مختلفة، و هي: العنونة بالمسجل – العنونة الفورية – العنونة المباشرة – العنونة غير المباشرة بالمسجل – العنونة القاعدية – العنونة المفهرسة - العنونة القاعدية المفهرسة – العنونة بالسلسلة – العنونة بالنافذة.

و هذه الأنظمة التسعة عدا العنونة بالمسجل و العنونة الفورية تتطلب الرجوع إلى المتحول المخزن في الذاكرة لذلك نحتاج لأن تبدأ وحدة ملائمة الممرات BIU بدورة ممر لقراءة أو كتابة في الذاكرة و هكذا فإن كل نظام عنونة له طريقة مختلفة لحساب عنوان المتحول الذي سيخرج على ممر العناوين أثناء دورة الممر، و سندرس الآن كلاً من هذه الأنظمة بالتفصيل:

ملاحظة: جميع التعليمات ستشرح لاحقاً.

أولاً: نظام العنونة بالمسجل

في هذا النظام يكمن المتحول بمسجل داخلي للمعالج، فمثلاً التعليمة التي تستعمل نظام العنونة بالمسجل هي MOV AX,BX و التي تعني نقل محتوى BX (متحول المصدر) إلى المسجل AX (متحول الهدف) أي أن تنفيذ هذه التعليمة يتم دون

الرجوع إلى الذاكرة أي في المعالج لأن كلا المسجلين AX و BX موجودين في المعالج:

نلاحظ من الشكلين السابقين و في الشكل الأول نجد أنه قد تم توليد العنوان الفيزيائي للتعليمة بواسطة الـIP و الـ CS حيث يتم إحضار التعليمة إلى المعالج و تتم فك شيفرتها (BC3٨ من الجدول) .

ثانياً: نظام العنونة الفورية

في هذا النظام يكون المتحول جزء من التعليمة و ليس مضمون سجل أو عنوان حجرة ذاكرة حيث يدعى هذا المتحول بالمتحول الفوري و المتحولات الفورية تمثل معطيات ثابتة يمكن أن تكون بايت أو كلمة (٢ بايت).

مثال: MOV AL,15 نجد أن متحول المصدر هو ١٥h و هو متحول مصدر فوري ذو بايت واحد و الشكلان التاليان يوضحان حالة المعالج قبل و بعد تنفيذ التعليمة السابقة.

ثالثاً: نظام العنونة المباشرة

يختلف هذا النظام عن نظام العنونة الفورية بأن الحجرات التي تلي رمز التعليمة تحوي على العنوان الفعال للذاكرة EA= Effective memory Address أي الإزاحة و هذا العنوان مؤلف من ١٦ بت حيث يتم توليد العنوان الفيزيائي انطلاقاً من DS و ES .

للأتصال/ mushtaq_talib58@yahoo.com

مثال:

[1234]MOV CX,

بفرض كان DS= 200 عندئذ العنوان الفيزيائي يحسب بالعلاقة :

03243h =1134 +200 x 10h =PA

ثم يذهب المعالج إلى الموقع ٠٣٢٣٤h في الذاكرة و يأخذ محتوى تلك الحجرة و يضعها في CL و يأخذ محتوى الحجرة التي تليها و يضعها في CH .

رابعاً: نظام العنونة غير المباشرة بالمسجل :

هذا النظام يشبه نظام العنونة المباشرة لكن يختلف عنه بأن العنوان الفعال (إزاحة) يكمن في مسجل مؤشر BX,BP أو مسجل دليل SI,DI .

مثال:

إن التعليمة التي تستخدم نظام العنونة غير المباشرة بالمسجل هي [SI]MOV AX, حيث يتم توليد العنوان الفيزيائي للمتحول بالاستناد إلى SI و DS . عن طريق العلاقة PA= (DS x 10h) + SI ، و بفرض كانت SI= 1234 و DS= 200 فإن PA= (0200 x 10) + 1234= 03234 و هو معامل المصدر حيث يذهب المعالج إلى الحجرة ٠٣٢٣٤ و يأخذ منها قيمتها و يضعها في AL أما قيمة الحجرة التي تليها فيتم وضعها في AH و يبين الشكل التالي حالة المعالج قبل و بعد تنفيذ التعليمة السابقة:

خامساً: نظام العنونة القاعدية

في هذا النظام من العنونة يحسب العنوان بواسطة جمع الإزاحة (disp)مع محتويات إما مسجل القاعدة BX أو مسجل مؤشر القاعدة BP مع القيمة الحالية الموجودة في المسجل DS أو SS على الترتيب أي:

disp +BP + (SS x 10h) =disp +BX + (DS X 10h) =PA

إن تعليمة MOV التي تستخدم العنونة القاعدية لتحديد حجرة متحول الهدف هي

Beta,AL.[BX]MOV

إن شيفرة التعليمة السابقة هي ٨٨٧٠ ٣٤١٢ و إن هذه التعليمة تستخدم مسجل القاعدة BX و الإزاحة المباشرة Beta لاشتقاق العنوان الفعال لمتحول الهدف حيث يتم تحقيق نظام العنونة القاعدية بواسطة تخصيص مسجل القاعدة أو مسجل مؤشر القاعدة بقوسين متوسطين (مربعين) متبوعاً بنقطة و إزاحة مباشرة (Beta). إن متحول المصدر في هذه التعليمة متوضع في البايت السفلي من المراكم أي في AL و بفرض أن قيمة Beta هي h١٢٣٤ فإن العنوان الفيزيائي لمتحول الهدف يتم حسابه بالعلاقة:

04234h =1234 +1000 +02000 =disp +BX + (DS x 10h) =PA

هذا العنوان الفيزيائي تحسبه الـ BIU و من ثم تطلب الـ EU بدء دورة ممر كتابة في الذاكرة و هكذا يكتب متحول المصدر AL في حجرة الذاكرة ذات العنوان الفيزيائي h٠٤٢٣٤ أي بعد تنفيذ التعليمة تصبح حالة المعالج كما هو واضح في الشكل السابق.

سادساً: نظام العنونة المفهرسة

في هذه الطريقة من العنونة يتم الحصول على العنوان الفعال نتيجة جمع محتوى مسجل الفهرس إما DI أو SI إلى عنوان الإزاحة displacement)disp) و هذا النوع من العنونة يناسب أغراض الجداول حيث يكون عنوان الإزاحة في بداية أول عنوان من الجدول و مسجل الفهرس يؤشر إلى أي عنصر من محتويات الجدول.

مثال: ليكن لدينا التعليمة التالية و التي شيفرتها A443412٨ و هي MOV [SI]AL,array .

هذه التعليمة يتم فيها تحديد متحول المصدر بواسطة العنونة المفهرسة المباشرة حيث أن array تمثل الإزاحة المباشرة و هي تسبق مسجل الدليل الموجود ضمن قوسين متوسطين، حيث يتم توليد العنوان الفيزيائي التالي:

EA + (DS x 10h) =PA

<= disp + (SI) =3234h ; EA =1234 +2000 =EA

05234h =3234 +02000 =EA + (DS x 10h) =PA

إن نتيجة تنفيذ هذه التعليمة هي أن محتويات حجرة الذاكرة ذات العنوان الفيزيائي ٠٥٢٣٤h تنقل إلى AL و تصبح حالة المعالج كما هو موضح في الشكل السابق.

سابعاً: نظام العنونة القاعدية المفهرسة :

في هذا النوع من العنونة يتم الحصول على العنوان الفعال نتيجة جمع محتوى مسجل القاعدة مع مسجل الفهرس و في حال وجود عنوان يجب إضافة هذا العنوان إلى محتوى المسجلين المذكورين.

مثال: التعليمة MOV AH, [BX].Beta[SI] يتم فيها حساب العنوان الفعال لمتحول المصدر كما يلي:

SI +Beta +BX =EA

06234h =4234 +02000 =EA + (DS x 10h) =PA

و يبين الشكل التالي حالة المعالج قبل و بعد تنفيذ التعليمة:

للأتصال/ mushtaq_talib58@yahoo.com

حيث نلاحظ أنه بعد تنفيذ التعليمة تصبح محتويات AH= BEh و التي تمثل محتويات حجرة الذاكرة ذات العنوان الفيزيائي ٠٦٢٣٤h .

ثامناً: نظام العنونة بالسلسلة

إن تعليمات السلسلة في مجموعة تعليمات المعالج ٨٠٨٦ تستعمل أوتوماتيكياً مسجل دليل المصدر و مسجل دليل الهدف لتعيين العناوين الفعالة لمتحولي المصدر و الهدف. فمثلاً تعليمة MOVS هي تعليمة النقل للسلسلة، و هي تستخدم SI و المقطع DS من أجل متحول المصدر و DI و المقطع ES من أجل متحول الهدف. و نلاحظ أنه لا SI و لا DI تظهران في تعليمة السلسلة.

تاسعاً: نظام العنونة بالنافذة

يستعمل هذا النظام مع تعليمات الإدخال و الإخراج لنوافذ I/O . من أجل النوافذ في حيِّز عنونة I/O يستخدم فقط نظام العنونة المباشرة و نظام العنونة غير المباشرة لاستعمال المسجل DX . فمثلاً العنونة المباشرة لنافذة دخل تكون كما في التعليمة التالية:

IN AL,15h ? IN AL,[15h]

للأتصال/ mushtaq_talib58@yahoo.com

تعني هذه التعليمة إدخال معطيات ذات بايت واحد من نافذة الدخل ذات العنوان ١٥h من حيز عنونة O/I إلى المسجل AL.

مثال آخر عن استعمال العنونة غير المباشرة للنافذة من أجل متحول المصدر هو التعليمة التالية:

[DX]IN AL,

هذا يعني إدخال معطيات ذات بايت واحد من نافذة الدخل التي عنوانها يكون محدد بواسطة مضمون مسجل DX فمثلاً: إذا كان DX= 1234h فإن محتويات النافذة ذات العنوان ١٢٣٤h يتم تحميلها في المسجل AL.

للأتصال/ mushtaq_talib58@yahoo.com